Expert Systems With Applications 46 (2016) 336–345

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Constrained min–max optimization via the improved constraint-activated differential evolution with escape vectors

Shu-Mei Guo[a,*], Pang-Han Hsu[a], Chin-Chang Yang[a], Jason Sheng-Hong Tsai[b]

[a] Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan 701, Taiwan, R.O.C.
[b] Control System Laboratory, Department of Electrical Engineering, National Cheng Kung University, Tainan 701, Taiwan, R.O.C.

ARTICLE INFO

Keywords:
Constrained min–max optimization
Differential evolution
Robust design

ABSTRACT

In system design, the best system designed under a simple experimental environment may not be suitable for application in real world if dramatic changes caused by uncertainties contained in the real world are considered. To deal with the problem caused by uncertainties, designers should try their best to get the most robust solution. The most robust solution can be obtained by constrained min–max optimization algorithms. In this paper, the scheme of generating escape vectors has been proposed to solve the problem of premature convergence of differential evolution. After applying the proposed scheme to the constrained min–max optimization algorithm, the performance of the algorithm could be greatly improved. To evaluate the performance of constrained min–max optimization algorithms, more complex test problems have also been proposed in this paper. Experimental results show that the improved constrained min–max optimization algorithm is able to achieve a quite satisfied success rate on all considered test problems under limited accuracy.

1. Introduction

Many real-world problems can be described by min–max optimization problems, such as robust design (Guo, Shieh, Chen, & Coleman, 2001, 2001b) and zero-sum game (Rapoport & Boebel, 1992). In robust design, uncertainties usually exist in our real world. A system which is well-designed under a deterministic scenario may perform poorly under other scenarios perturbed by some uncertainties. Therefore, a designer needs to design a system to achieve the best possible performance in the worst case scenario, i.e. a solution with high tolerance to uncertainties.

A more specific real application can be seen in the robust design of Kalman filtering (Guo et al., 2001, 2001b). More precisely, a quite basic min–max evolutionary programming (EP) (compared to the state-of-art methodology newly proposed in this paper) was proposed by Guo et al. to construct the “best” nominal Kalman filter for the nonlinear discrete-time stochastic systems with unknown-but-bounded system parameters as well as stochastic system and measurement noises. The worst-case realization of the uncertain discrete-time nonlinear stochastic system with respect to the determined “best” nominal filter is also given in (Guo et al., 2001, 2001b). The physical interpretation of this application is that based on the implemented “best” nominal filter determined by the min–max principle optimization, no realization of the uncertain discrete-time nonlinear stochastic system can induce a worse filtering performance than the above-determined worst-case realization. As a result, if the min–max optimization-based worst case performance can be accepted under the pre-specified control specification, then the “best” nominal filter guarantees to induce an even better performs for any other realization of this uncertain system.

Evolutionary algorithms (EAs), such as genetic algorithm (GA), particle swarm optimization (PSO) and differential evolution (DE), have been successfully applied to many scientific optimization problems, engineering applications and image processing (Guo & Yang, 2011b; Guo, Lai, Chou, & Yang, 2011a). This is because the EAs have certain advantages over the conventional optimization methods. When faced with the problems which are non-linear or non-differentiable, conventional methods, e.g. the Karush–Kuhn–Tucker (KKT) based method, confront some troubles, such as availability of gradient information. In contrast, the EAs do not introduce this kind of troubles. In 1995, Storn and Price proposed differential evolution (DE) (Price, Storn, & Lampinen, 2006; Storn & Price, 1995, 1997), which features simplicity and efficiency. Later, emergences of many methods (Brest, Greiner, Bošković, Mernik, & Zumer, 2006; Islam, Das, Ghosh, Roy, & Suganthan, 2012; Qin, Huang & Suganthan, 2009; Zhang & Sanderson, 2009) over the past two decades have greatly improved the performance of original DE. In the last years, DE has been widely used in various domains of real-world

* Corresponding author. Tel.: +886 6 275 7575.
E-mail addresses: guosm@mail.ncku.edu.tw (S.-M. Guo), p76021093@mail.ncku.edu.tw (P.-H. Hsu), p78981061@mail.ncku.edu.tw (C.-C. Yang), shtsai@mail.ncku.edu.tw (J.S.-H. Tsai).

http://dx.doi.org/10.1016/j.eswa.2015.10.042
0957-4174/

optimization problems, such as folded laminated composite plates (Le-Anh, Nguyen-Thoi, Ho-Huu, Dang-Trung, & Bui-Xuan, 2015) and layout optimization of truss structures (Ho-Huu, Nguyen-Thoi, Nguyen-Thoi, & Le-Anh, 2015).

In the literature, the algorithms proposed in (Basar et al., 1995; Cramer, Sudhoff, & Zivi, 2009; Herrmann, 1999; Hur, Lee, & Tahk, 2003; Jensen, 2004; Shi & Krohling, 2002) have been able to solve unconstrained min–max optimization problems. Nonetheless, these algorithms are developed without taking constraints into account, so they are unable to solve constrained min–max optimization problems. To solve constrained min–max optimization problems, the KKT condition based min–max optimization algorithm (Lu, Cao, jie Yuan, & Zhou, 2008) was proposed, but it is not applicable to many real world problems due to the assumptions of continuity and convexity on objective function and availability of Hessian information. Sequential differential evolution (SDE) (Segundo, Krohling, & Cosme, 2012) has shown outstanding performance with a high success rate on the considered constrained min–max optimization problems. Nevertheless, the known solutions of the problems are not precise enough, which cannot judge the accuracy of solutions the algorithms can achieve. Thanks to constraint-activated differential evolution (CaDE) (Guo, Yang, Chang, & Tsai, 2015), we can obtain more accurate solutions. CaDE improves the performance with three proposed components, and experimental results show that it is better than SDE. However, the number of test problems is still too few to prove its robustness. Some operations of CaDE cause the algorithm to run a risk of premature convergence so that it can hardly find global optima in complex min–max test problems.

In this paper, we propose more complex min–max test problems to assess the performance of CaDE. Nevertheless, a scheme of generating escape vectors is proposed in this paper to deal with this kind of problems. Escape vectors are able to guide the population to escape from local optima by increasing the population diversity to prevent premature convergence.

The rest of this paper is structured as follows. In Section 2, the definition of constrained min–max optimization problem is described. A brief introduction of differential evolution and the algorithm of CaDE are discussed. We also discuss the shortcoming of CaDE. Section 3 presents the proposed method. In Section 4, the test problems are discussed. In Section 5, experimental results are presented. Concludes are given in Section 6.

2. Background

In this section, the definition of constrained min–max optimization problem is described. The DE algorithm and CaDE algorithm are presented and the shortcoming of CaDE is also discussed in detail to clarify the related background.

2.1. Definition of constrained min–max optimization problem

A constrained min–max optimization problem can be stated as follows

$$\min_{\vec{x}\in X}\max_{\vec{y}\in Y} f(\vec{x},\vec{y}) \tag{1}$$

subject to

$$g_i(\vec{x},\vec{y}) \le 0, \quad i = 1, 2, 3 \dots n_c$$

where $f(\vec{x},\vec{y})$ is an objective function, $X \subset \mathbb{R}^n$, $Y \subset \mathbb{R}^m$, X and Y are the sets defining the search space of the problem, $\vec{x} = [x_1, x_2, x_3, \dots, x_n]^T$ and $\vec{y} = [y_1, y_2, y_3, \dots, y_m]^T$ are parameter vectors, $g_i(\vec{x},\vec{y})$ is the ith inequality constraints, n_c is the number of inequality constraint.

However, a greater-than inequality $g(\vec{x},\vec{y}) \ge 0$ can be transformed into a less-than inequality by $-g(\vec{x},\vec{y}) \le 0$ and an equality constraint $h(\vec{x},\vec{y}) = 0$ can also be replaced with two following inequalities

$$\begin{cases} h(\vec{x},\vec{y}) \le 0 \\ -h(\vec{x},\vec{y}) \le 0 \end{cases}. \tag{2}$$

Without loss of generality, we consider only equal to and less-than inequality constraints.

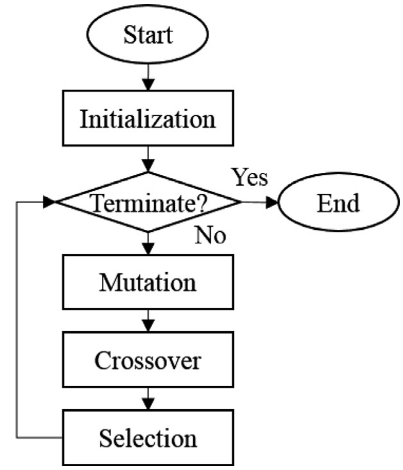

Fig. 1. Flowchart of DE algorithm.

2.2. Differential evolution with constraint handling

DE is an effective population-based global optimizer and one of EAs. It evolves a population towards to a global optimum by a series of operations, i.e. initialization, mutation, crossover and selection. The flowchart of DE algorithm is shown in Fig. 1.

There is a notation to express different schemes of DE: DE/x/y/z, where “DE” represents differential evolution, “x” is a string showing the composition of a base vector, “y” stands for the number of differences to perturb a base vector, and “z” indicates the type of crossover used. There are two types of crossover operators widely used in the literature: “exp” and “bin”. They represent for exponential crossover and binominal crossover. In this paper, we employ binominal crossover since it performs better than the exponential crossover (Efren, Jesus, & Carlos, 2006).

2.2.1. Initialization

The first step of DE is to randomly initialize NP D-dimensional real-valued vectors (individuals) so as to form a population

$$P_G = \left\{\vec{x}_{i,G} | \vec{x}_{i,G} = [x_{1,i,G}, x_{2,i,G}, x_{3,i,G}, ..., x_{D,i,G}]^T, \quad i = 1, 2, 3, ..., NP\right\} \tag{3}$$

where G represents the generation number, $\vec{x}_{i,G}$ is the ith individual in the population P_G with components $x_{j,i,G}$, $j = 1, 2, 3, \dots, D$. The individuals in the initial population P_0 are uniformly distributed over the entire search space between upper bounds $\vec{x}_{\max}$ and lower bounds $\vec{x}_{\min}$

$$\vec{x}_{\max} = [x_{1,\max}, x_{2,\max}, \dots, x_{D,\max}]^T \tag{4}$$

$$\vec{x}_{\min} = [x_{1,\min}, x_{2,\min}, \dots, x_{D,\min}]^T. \tag{5}$$

We can obtain the value of the jth component of the ith initial individual according to

$$x_{j,i,G} = x_{j,\min} + rand_{j,i}[0, 1] \cdot (x_{j,\max} - x_{j,\min}) \tag{6}$$

where $rand_{j,\,i}[0, 1]$ is a uniformly distributed random number in the range between 0 and 1.